**一、引言**  
  
          Redis的基本数据类型，高级特性，与Lua脚本的整合等相关知识点都学完了，说是学完了，只是完成了当前的学习计划，在以后的时间还需继续深入研究和学习。从今天开始来讲一下有关Redis的集群模式，Redis有三种集群模式，第一个就是主从模式，第二种“哨兵”模式，第三种是Cluster集群模式，第三种的集群模式是在Redis 3.x以后的版本才增加进来的，我们今天就来说一下Redis第一种集群模式：主从集群模式。  
  
**二、配置操作**  
  
         实现主从复制（Master-Slave Replication）的工作原理：Slave从节点服务启动并连接到Master之后，它将主动发送一个SYNC命令。Master服务主节点收到同步命令后将启动后台存盘进程，同时收集所有接收到的用于修改数据集的命令，在后台进程执行完毕后，Master将传送整个数据库文件到Slave，以完成一次完全同步。而Slave从节点服务在接收到数据库文件数据之后将其存盘并加载到内存中。此后，Master主节点继续将所有已经收集到的修改命令，和新的修改命令依次传送给Slaves，Slave将在本次执行这些数据修改命令，从而达到最终的数据同步。  
  
        如果Master和Slave之间的链接出现断连现象，Slave可以自动重连Master，但是在连接成功之后，一次完全同步将被自动执行。  
  
       我先介绍一下我的环境，操作系统是Windows 10企业版，Redis有两个版本，第一版本是以Windows服务形式安装的Redis服务器，IP地址是：192.168.131.1，端口号：6379，当前没有设置密码；第二个版本是在Linux系统下安装的Redis服务，IP地址是：192.168.127.128，端口号是：6379，当前也没有设置密码。今天测试两个情况，一种情况：Windows系统上的Redis服务做主服务节点，Linux系统上的Redis服务做从服务节点，第二种情况是：Linux系统上的Redis服务做主服务节点，Windows系统上的Redis服务做从服务节点。其他情况就不测试了，同系统之间测试就很容易了，也不会出现什么问题。  
  
       主从复制配置：  
  
       第一步：修改从节点的配置文件：slaveof <masterip> <masterport>  
  
       第二步：如果设置了密码，就要设置：masterauth <master-password>  
  
       主从复制的配置很简单，主要操作从节点的配置文件，主节点不需要任何改动。我们可以使用info查看role角色即可知道是主服务或从服务。  
  
  
**1、Windows系统上的Redis服务做主服务节点，Linux系统上的Redis服务做从服务节点（测试很顺利）**

[IMG_256](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//主节点服务：192.168.131.1 端口号：6379 Windows系统

//从节点服务：192.168.127.128 端口号：6379 Linux系统

//现在主要修改从节点服务Linux系统上的redis.conf配置文件

slaveof 192.168.131.1 6379

设置完成:

//主节点配置信息：

192.168.131.1:6379>info Replication

#Replication

role:master

connected\_slaves:1

slave0:ip=192.168.131.1,port=6379,state=online,offset=239,lag=1

master\_repl\_offset:239

repl\_backlog\_active:1

repl\_backlog\_size:1048576

repl\_backlog\_first\_byte\_offset:2

repl\_backlog\_histlen:238

//从节点配置信息：

192.168.127.128:6379>info Replication

#Replication

role:slave

master\_host:192.168.131.1

master\_port:6379

master\_link\_status:up

master\_last\_io\_seconds\_ago:8

master\_sync\_in\_progress:0

slave\_repl\_offset:253

slave\_priority:100

slave\_read\_only:1

connected\_slaves:0

master\_replid:7f2e5cde55803c8b78d26c16f0111695e3c1fb6f8

master\_replid2:000000000000000000000000000000000000000000

master\_repl\_offset:253

second\_repl\_offset:-1

repl\_backlog\_active:1

repl\_backlog\_size:1048576

repl\_backlog\_first\_byte\_offset:2

repl\_backlog\_histlen:252

[IMG_257](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

       2、Linux系统上的Redis服务做主服务节点，Windows系统上的Redis服务做从服务节点（设置完成，但是主节点不能连接，master\_link\_status:down，该问题还没解决）

[IMG_258](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//主节点服务：192.168.127.128 端口号：6379 Linux系统

// 从节点服务：192.168.131.1 端口号：6379 Windows系统

//现在主要修改从节点服务在Windows系统上的redis.windows.conf配置文件

slaveof 192.168.127.128 6379

//如果需要密码

//masterauth 123456

//主节点配置信息：

192.168.127.128:6379>info Replication

#Replication

role:master

connected\_slaves:0

master\_replid:23ed05016a5fdf45e45318281b7f827cbbf75025

master\_replid2:0000000000000000000000000000000000000000

master\_repl\_offset:0

second\_repl\_offset:-1

repl\_backlog\_active:1

repl\_backlog\_size:1048576

repl\_backlog\_first\_byte\_offset:1

repl\_backlog\_histlen:0

//从节点配置信息：

192.168.127.128:6379>info Replication

#Replication

role:slave

master\_host:192.168.127.128

master\_port:6379

master\_link\_status:down

master\_last\_io\_seconds\_ago:-1

master\_sync\_in\_progress:0

slave\_repl\_offset:1

master\_link\_down\_since\_seconds:jd

slave\_priority:100

slave\_read\_only:1

connected\_slaves:0

repl\_backlog\_active:0

repl\_backlog\_size:1048576

repl\_backlog\_first\_byte\_offset:0

repl\_backlog\_histlen:0

[IMG_259](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

**Slave从节点日志显示为：**

[IMG_260](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

\* Connecting to MASTER 192.168.127.128:6379[11188] 07 Feb 15:33:10.908

\* MASTER <-> SLAVE sync started[11188] 07 Feb 15:33:10.909

\* Non blocking connect for SYNC fired the event.[11188] 07 Feb 15:33:10.913

\* Master replied to PING, replication can continue...[11188] 07 Feb 15:33:10.917

\* Partial resynchronization not possible (no cached master)[11188] 07 Feb 15:33:10.923

\* Full resync from master: 0e17ac45471c6a94dadec46f993c14ee6dc33726:0[11188] 07 Feb 15:33:10.980

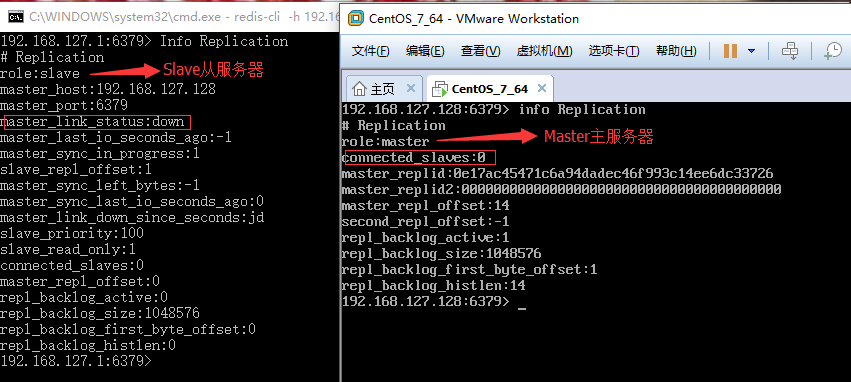
\* MASTER <-> SLAVE sync: receiving 193 bytes from master[11188] 07 Feb 15:33:10.987

\* MASTER <-> SLAVE sync: Flushing old data[11188] 07 Feb 15:33:10.989

**\* MASTER <-> SLAVE sync: Loading DB in memory[11188] 07 Feb 15:33:10.991 # Can't handle RDB format version 8[11188] 07 Feb 15:33:10.992**

**# Failed trying to load the MASTER synchronization DB from disk[11188] 07 Feb 15:33:11.910 红色字体，就是问题所在，目前自己能力有限，还未解决**

[IMG_261](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

   效果截图如下：  
  
                 

             有谁可以解决问题的，请给我留言，不胜感谢，我自己也会继续研究的  
  
**三、主从模式的配置**  
  
      下面是我使用的配置，需要修改的配置项我写了出来，没有更改的配置项就是用默认值，就不会写出来：  
  
**1、######Master config**  
  
**1.1、### NETWORK**

[IMG_263](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

bind 192.168.127.128

port 6379

timeout 30 # Client 端空闲断开连接的时间

daemonize yes #默认值是no，把值修改为yes，以后台模式运行

[IMG_264](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

**1.2、### GENERAL**

logfile /root/application/program/redis-tool/logs/redis.log #日志文件的位置

**1.3、### SNAPSHOTTING 设置：**

dir /root/application/program/redis-tool/datas #SNAPSHOTTING文件的路径

**1.4、### APPEND ONLY MODE 设置**

appendonly yes #默认值是No，意思是不使用AOF增量持久化的方式，使用RDB全量持久化的方式。把No值改成Yes，使用AOF增量持久化的方式

appendfsync always

**2、###### Slave Config**  
  
**2.1、### NETWORK**

[IMG_265](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

bind 192.168.127.129

port 6379

timeout 30 # Client 端空闲断开连接的时间

daemonize yes #默认值是no，把值修改为yes，以后台模式运行

[IMG_266](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

**2.2、### GENERAL**

logfile /root/application/program/redis/logs/redis.log #日志文件的位置

**2.3、### SNAPSHOTTING 设置：**

dir /root/application/program/redis/datas #SNAPSHOTTING文件的路径

**2.4、### REPLICATION 设置：**

slaveof 192.168.127.128 6379

slave-serve-stale-data no #如果slave 无法与master 同步，设置成slave不可读，方便监控脚本发现问题。

**2.5、### APPEND ONLY MODE 设置：**

appendonly yes #默认值是No，意思是不使用AOF增量持久化的方式，使用RDB全量持久化的方式。把No值改成Yes，使用AOF增量持久化的方式

appendfsync always

**3、用redis-cli bgsave 命令每天凌晨一次持久化一次master redis上的数据，并CP到其它备份服务器上。**  
  
**4、用redis-cli bgrewriteaof 命令每半小时持久化一次 slave redis上的数据，并CP到其它备份服务器上。**  
  
**5、写个脚本 ，定期get master和slave上的key,看两个是否同步，如果没有同步，及时报警。**  
  
  
**四、主从模式的优缺点**  
  
**1、Redis的Replication的特点和优点：**  
  
                  1】、同一个Master可以同步多个Slaves。  
  
                  2】、Slave同样可以接受其它Slaves的连接和同步请求，这样可以有效的分载Master的同步压力。因此我们可以将Redis的Replication架构视为图结构。  
  
                  3】、Master Server是以非阻塞的方式为Slaves提供服务。所以在Master-Slave同步期间，客户端仍然可以提交查询或修改请求。  
  
                  4】、Slave Server同样是以非阻塞的方式完成数据同步。在同步期间，如果有客户端提交查询请求，Redis则返回同步之前的数据。  
  
                  5】、为了分载Master的读操作压力，Slave服务器可以为客户端提供只读操作的服务，写服务仍然必须由Master来完成。即便如此，系统的伸缩性还是得到了很大的提高。  
  
                  6】、Master可以将数据保存操作交给Slaves完成，从而避免了在Master中要有独立的进程来完成此操作。  
  
                  7】、支持主从复制，主机会自动将数据同步到从机，可以进行读写分离。  
  
**2、Redis的Replication的缺点：**  
  
                1】、Redis不具备自动容错和恢复功能，主机从机的宕机都会导致前端部分读写请求失败，需要等待机器重启或者手动切换前端的IP才能恢复。  
  
                2】、主机宕机，宕机前有部分数据未能及时同步到从机，切换IP后还会引入数据不一致的问题，降低了系统的可用性。  
  
                3】、Redis的主从复制采用全量复制，复制过程中主机会fork出一个子进程对内存做一份快照，并将子进程的内存快照保存为文件发送给从机，这一过程需要确保主机有足够多的空余内存。若快照文件较大，对集群的服务能力会产生较大的影响，而且复制过程是在从机新加入集群或者从机和主机网络断开重连时都会进行，也就是网络波动都会造成主机和从机间的一次全量的数据复制，这对实际的系统运营造成了不小的麻烦。  
  
                4】、Redis较难支持在线扩容，在集群容量达到上限时在线扩容会变得很复杂。为避免这一问题，运维人员在系统上线时必须确保有足够的空间，这对资源造成了很大的浪费。  
  
**四、结束**  
  
           今天就写到这里了，其实redis的主从模式很简单，在实际的生产环境中是很少使用的，我也不建议在实际的生产环境中使用主从模式来提供系统的高可用性，之所以不建议使用都是由它的缺点造成的，在数据量非常大的情况，或者对系统的高可用性要求很高的情况下，主从模式也是不稳定的。虽然这个模式很简单，但是这个模式是其他模式的基础，所以必须深刻的理解，对其他模式的学习才会有帮助作用。